University of Toronto (Mississauga Campus)

CSC411- Machine Learning and Data Mining

Neural Network Toolbox in Matlab

Tutorial 4 – Feb 9th, 2007

Basic Neural Network Toolbox Flow Diagram

Network Structure Creation

Matlab Command: *newlin, newp, newff*

↓

Training the Network

Matlab Command*: train or adapt*

↓

Simulation

Matlab Command*: sim*

Neural Network Structure – newlin, newp, and newff

	newlin	**newp**	**newff**
Description	Create a linear layer	Create a perceptron	Create a feed-forward backpropagation network
model	net=newlin(PR,S,ID,LR)	Net=newp(PR,S,TF,LF)	Net=newff(PR,[S1 S2…SN],{TF1 TF2…TFN},BTF, BLF, PF)
PR	RX2 matrix of min and max values for R input elements		
S / Si	S: Number of elements (neurons) in the output vector		Si: Size of ith layer
ID	Input delay vector, **default = [0]**	--	--
LR	Learning rate, **default = 0.01**	--	--
TF	--	Transfer function: (**hardlim** or hardlims)	TFi: Transfer function at ith layer (purelin, logsig or **tansig**)
BTF	--	--	BTF: Backprop network training function: (**trainlm**, trainbfg, trainrp, traingd or traingdx)
LF/BLF	--	LF: Learning function: (**learnp** or learnpn)	BLF: Backprop weight/bias learning function: (learngd or **learngdm**)
PF	Performance measure: mse	Performance measure: mae	Performance measure: (**mse** or mae)

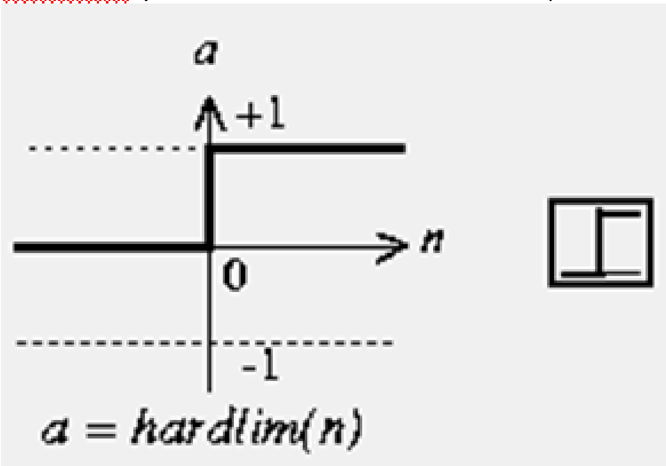
Neural Network Structure – newlin, newp, and newff

	newlin	newp	newff
TF	--	Transfer function: (**hardlim** or hardlims)	TFi: Transfer function at ith layer (purelin, logsig or **tansig**)

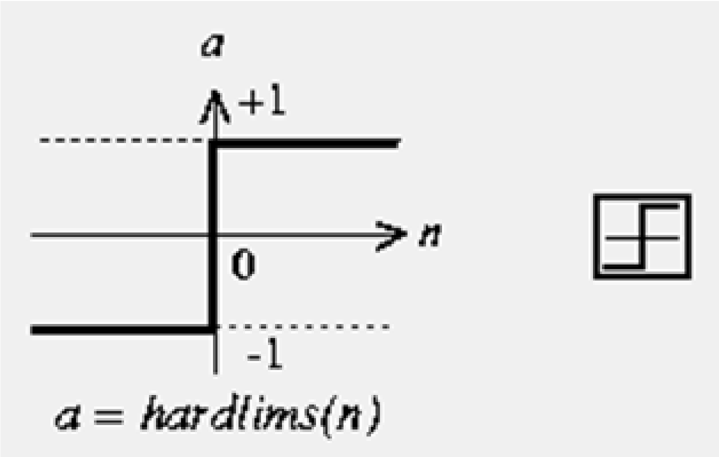
Transfer functions:

Used in newp:

hardlim (Hard Limit Trans Function)



hardlims(Symmetric Hard Lim Trans Function)

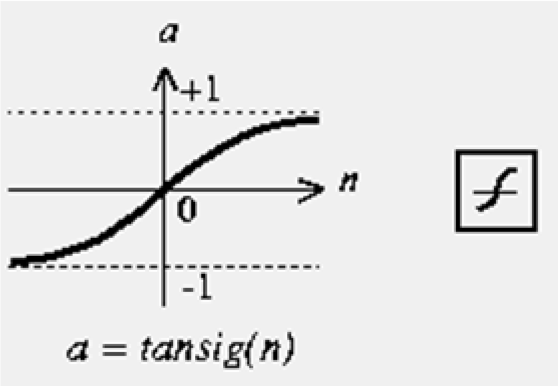


Neural Network Structure – newlin, newp, and newff

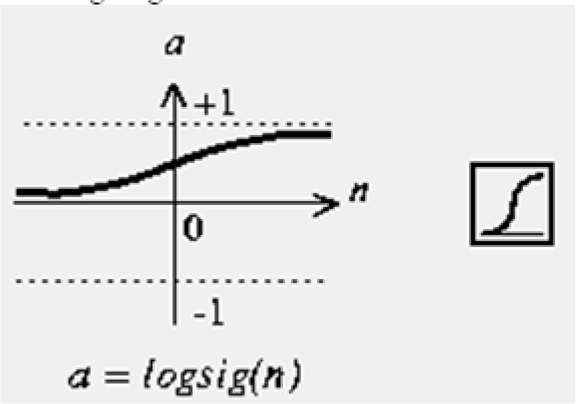
	newlin	newp	newff
TF	--	Transfer function: (**hardlim** or hardlims)	TFi: Transfer function at ith layer (purelin, logsig or **tansig**)

Used in newff:

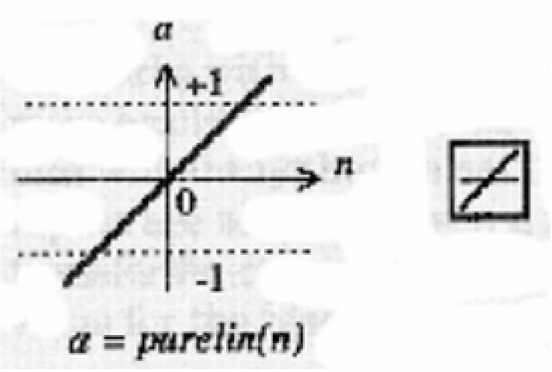
Tan-Sigmoid Tran Function



Log-Sigmoid Tran Function

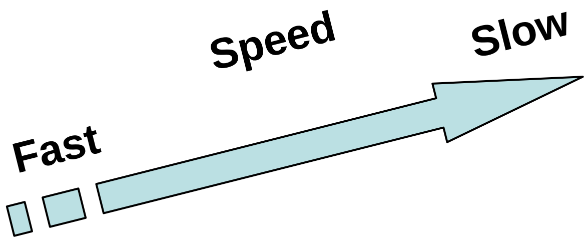


Linear Tran Function

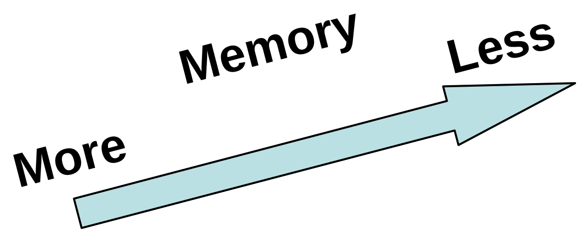


Neural Network Structure – newlin, newp, and newff

	newlin	**newp**	**newff**
Description	Create a linear layer	Create a perceptron	Create a feed-forward backpropagation network
BTF	--	--	BTF: Backprop network training function: (**trainlm**, trainbfg, trainrp, traingd or traingdx)



trainlm → trainbfg→trainrp



Neural Network Structure – newlin, newp, and newff

	newlin	**newp**	**newff**
Description	Create a linear layer	Create a perceptron	Create a feed-forward backpropagation network
LF/BLF	--	LF: Learning function: (**learnp** or learnpn)	BLF: Backprop weight/bias learning function: (learngd or **learngdm**)
PF	Performance measure: mse	Performance measure: mae	Performance measure: (**mse** or mae)

Learning Function: learnp or learnpn

If input vectors have a large variance in their lengths, the *learnpn* can be faster than *learnp*

Performance Function: mse or mae

mse: Mean squared error performance function

mae: Mean absolute error performance function

Basic Neural Network Toolbox Flow Diagram

Network Structure Creation

Matlab Command: *newlin, newp, newff*

↓

Training the Network

Matlab Command*: train or adapt*

↓

Simulation

Matlab Command*: sim*

Neural Network Training: adapt and train

	Adapt	Train
Syntax	**[net**,Y,E,Pf,Af]=adapt(**net, P**,T,Pi,Ai)	**[net**, tr]=train(**net,P,T**,Pi,Ai)
Parameters	Net: network P: Network inputs T: Network targets, default = 0 (optional in adapt) Pi: Initial input delay conditions, default =0 (optional) Ai: Initial layer delay conditions, default =0 (optional)	
Returns	Net: new network Y: Network outputs E: Network Errors Pf: Final input delay conditions Af: Final layer delay conditions	Net: new network Tr: Training record (i.e., epoch)
Related concepts	Net.adaptFcn='trains'	Net.trainFcn='trainb' : batch training Net.trainFcn ='trainc': online training
Conclusions	• Adapt takes more time then Train • Train provides more choice in training functions (gradient descent, Levenberg-Marquardt, etc) • For static networks, Usually train is a better choice	

Basic Neural Network Toolbox Flow Diagram

Network Structure Creation

Matlab Command: *newlin, newp, newff*

↓

Training the Network

Matlab Command*: train or adapt*

↓

Simulation

Matlab Command*: sim*

Neural Network Simulation: sim

`sim` simulates neural networks.

`[Y,Pf,Af] = sim(net,P,Pi,Ai)` takes,

- `net` – Network.
- `P` – Network inputs.
- `Pi` – Initial input delay conditions, default = zeros.
- `Ai` – Initial layer delay conditions, default = zeros.

and returns,

- `Y` – Network outputs.
- `Pf` – Final input delay conditions.
- `Af` – Final layer delay conditions.

Example – Rewrite XOR Problem

XOR

I_1	I_2	Out
0	0	0
0	1	1
1	0	1
1	1	0

```
%Input Data P
P = [1 1 0 0; 1 0 1 0];
%Target Data T
T = [0 1 1 0];
```

Network Structure Creation

Matlab Command: *newlin, newp, newff*

```
%Construct the Neural Network
%net = newff(PR,[S1..Sn],{TF1 ...TFn},BTF, BLF, PF);
net=newff([0 1; 0 1], [2 1],{'tansig' 'purelin'},'trainlm');
```

Training the Network

Matlab Command: *train or adapt*

```
%Update the parameters for the training
net.trainParam.epochs=10000;
net.trainParam.show=5;

%Train the neural network
net=train(net,P,T);
```

Simulation

Matlab Command: *sim*

```
%Simulate the neural network
Y=sim(net,P);
```

References:

- Matlab Help Desk: http://www-ccs.ucsd.edu/matlab/helpdesk.html
- Neural Network Toolbox in Matlab 2006: http://www.control.hut.fi/Kurssit/AS-74.3115 /Materiaali /Material2007/ Neural_Network_Toolbox_Slides.pdf
- Neural Network Toolbox: A tutorial for the Course Computational Intelligence: http://www.igi.tugraz.at/lehre/EW/tutorials/nnt_intro/nnt_intro.pdf